L3: Entry 13 of 29 File: USPT Aug 21, 2001

DOCUMENT-IDENTIFIER: US 6279051 B1
TITLE: Host adapter having paged payload buffers for simultaneously transferring data between a computer bus and a peripheral bus

Brief Summary Text (12):
The data transfer described above can be initiated by transferring to host adapter 12 a command in the form of a "Sequencer Control Block" (SCB) that contains information needed by host adapter 12 to perform the data transfer, as described by Stuber et al. at column 17, line 66 et. seq. Moreover, host adapter 12 can transfer the data to/from system memory 64 via a scatter/gather mechanism that stores the data in a number of portions in system memory 64. The SCB includes "a pointer to a scatter/gather data transfer pointer list, [and] a count of the number of elements in the scatter/gather list" (column 18, lines 3-5) that together indicate the portions of system memory 64 (FIG 1A) to or from which the data is to be transferred.

Detailed Description Text (7):
Host adapter 112 includes, in accordance with the invention, a data transfer module 114 that couples each of receive interface module 123 and send interface module 124 to host interface module 117. Data transfer module 114 includes a receive data path 118 for processing (in one implementation in a first-in-first-out manner) data received in one or more messages from peripheral bus 140 (via serializer-deserializer 113 and receive interface module 123), and transmission of the data to host interface module 117. Data transfer module 114 also includes a send data path 119 for processing data (in one implementation in a first-in-first-out manner) from host interface module 117 for transmission of the data in one or more messages onto peripheral bus 140 (via send interface module 124 and serializer-deserializer 113). Data transfer module 114 accesses system memory 164 via computer 120 at addresses defined in the scatter-gather lists. Host adapter 112 also includes a sequencer module 115 that controls and coordinates the transfer of data between computer 120 and peripheral bus 140. Sequencer module 115 is implemented by storage elements (not shown) and a RISC processor 150 (FIG. 2B) that is programmed with instructions to implement a multi-tasking protocol engine. Sequencer module 115 communicates with various modules, e.g. data transfer module 114 and host interface module 117 via an internal bus 121 (such as the CIO bus described in U.S. Pat. No. 5,659,690, or as described in U.S. patent application Ser. No. 09/088,812.

Detailed Description Text (8):
In one example, host processor 161 prepares a TCB that indicates a command to read data from a device 131. On receiving the TCB, host adapter 112 sends the command to device 131, and device 131 responds by sending data. On receipt of a message from peripheral bus 140, receive data path 118 (FIG. 2A) temporarily stores the header to determine the destination in system memory 164 to which an associated payload is to be transferred. Specifically, in one implementation, receive data path 118 uses a predetermined field in the header (e.g. the sequence identifier of a Fibre Channel frame) to identify the just-described TCB that is associated with the received message. The TCB indicates a scatter gather element (as described in, for example, U.S. Pat. No. 5,659,690) that in turn contains an address in system memory 164 to which the payload is to be transferred. So, sequencer module 115 loads an address and a count (from the scatter gather element) into registers 117C (e.g. an

address register and a count register) in host interface module 117. Thereafter, host interface module automatically transfers the data from receive data path 118 to system memory 164.

Detailed Description Text (13):
Sequencer module 115 polls (as illustrated by act 321 in FIG. 3D) various registers in the various modules, including the event flag in RFC 210, and performs various actions depending on the type of mismatch. For example, if the header received by RFC 210 has the same value in the TCB field, then sequencer module 115 drives appropriate signals to other modules, e.g. to SFC 119 to close a sequence (as defined in the Fibre Channel protocol). As another example, if the header received by RFC 210 has a different value in the TCB field, then sequencer module 115 determines that a context switch has occured, and performs any related operations. For example, sequencer module 115 may copy a TCB currently being used by RFC 210 to adapter memory 111 (FIG. 2A) for further processing at a later time. Simultaneous with such operations of sequencer module 115, RFC switches context and starts receiving messages from another peripheral device (e.g. device 132 in FIG. 2A). Sequencer module 115 also updates values in various modules, e.g. stores new scatter-gather values in registers 117N (FIG. 2B) of host interface module 117, and new values in expected header buffer of RFC 210. Sequencer module 115 obtains the new scatter-gather values from a new TCB that is retrieved from adapter memory 111 using the sequence identifier from the just-received header as a pointer to identify the new TCB. After updating of the expected header buffer, RFC 210 performs another comparison between the expected header buffer and the just-received header, and another mismatch indicates an error. If there is a match, sequencer module 115 sets up (as illustrated by act 323 in FIG. 3D) various registers for transmission (e.g. by DMA) of the payload associated with the received header to system memory 164 (FIG. 2A). Specifically, sequencer module 115 loads an address and a count (from the scatter gather element associated with the TCB) into registers 117N (e.g. an address register and a count register) to be used during transmission of the just-described payload. Such operations of sequencer module 115 are not critical aspects of the invention

Detailed Description Text (16):
Specifically, in response to the context done signal being active, host interface module 117 stops using a first group of storage elements 117C and starts using a second group of storage elements 117N that contain the destination address for the currently-received data to be transferred in the next cycle. In one specific implementation, each of groups 117C and 117N includes a scatter gather element that is held in two storage elements: a first storage element to hold an address in system memory 164, and a second storage element to hold a count indicating the amount of data to be transferred to system memory 164. Note that host interface module 117 begins using second group 117N even in the absence of a context done signal, e.g. when a total number of bytes indicated in the count of the first group 117C have been transferred. When beginning to use second group 117N, host interface module 117 informs sequencer module 115 that a new scatter gather element needs to be loaded into first group 117C. In response, sequencer module 115 (FIG. 2B) updates the values in first group 117C for use by host interface module 117 in future e.g. when the amount of data indicated by the count in second group 117N has been transferred (or at the end of another context). Therefore, host interface module 117 transfers data to system memory 164 (FIG. 2A) in a continuous manner without any delay between the use of storage elements in groups 117C and 117N. A host interface module 117 having more than one group (in this example two groups 117C and 117N) as discussed herein has a headstart when transitioning between scatter gather elements, due to elimination of a delay in determining and loading a new scatter gather element (i.e. delay between completion of the use of a scatter gather element, and beginning the use of a new scatter gather element) that is otherwise present in the prior art.

Detailed Description Text (34):

RFC 210, after comparing buffer H3 with the expected header buffer, finds a mismatch, raises an event for sequencer module 115, and sequencer module 115 updates the expected header buffer. Note that during this time, as pages 221A and 221B are emptied, RFC 210 drives a control signal to SFC 230 to send a receiver ready message on peripheral bus 140, each time a page becomes available (because in this example, each payload requires only one page). When RFC 210 drives the update acknowledge signal, the end-of-context signal RFC_EOF_CTX is active, and RPB 220 responds by clearing the end of payload flag CI, but flag DI stays set. Therefore, when all data from page 221B has been transferred, RPB 220 drives a context done signal (e.g. signal RPB_CTX_DONE) to indicate to host interface module 117 that there is no more data available for this context. In response to the context done signal, module 117 raises an event for the sequencer module 115, and module 115 checks if the scatter gather list is at the end (i.e. that all data to be transferred has been transferred). If not, sequencer module 115 saves the values in various registers for use at a later time to complete the data transfer, and clears the end of context flag DI in page 221B . In response to flag DI being cleared, RPB 220 starts transferring data from the next page 221I, and again drives an update request signal (e.g. signal RPB_RSP_REQ) to RFC 210 on bus 228.